Charity Link

Team 23:
Natalie Harrison
Varun Jasti
Aanya Jha
Tianze Wu (Susan)

Purpose

The underprivileged often have to seek out resources for themselves, which at times is not easy, efficient, or secure for them to carry out. Donations are popularized by Goodwill, and even those are often resold for a higher value price. Our project aims to streamline store management and distribution of outreach programs held by non-profit organizations, along with providing a user-friendly donation platform for individuals as well.

For people who are in need, our project’s purpose is to give them a one-stop location for all their food, clothing and various other needs online. For non-profit organizations, our project’s purpose is to create an efficient and useful platform for them to utilize when managing stock and delivering items.

Functional Requirements

User:

1. User Accounts
    - As a user, I would like to create an account.
    - As an organization, I would like to register my organization/create an account.
    - As a user, I would like to log into my account securely.
    - As an organization, I would like to log into my account securely.
    - As an organization, I would like to create organization manager accounts.
    - As an organization manager, I would like to log into my account securely.
    - As a user, I would like to log out of my account securely.
    - As an organization, I would like to log out of my account securely.
    - As an organization manager, I would like to log out of

my account securely.
- As a user, I would like to delete my account.
- As an organization, I would like to be able to delete my manager accounts.
- As an organization, I would like to be able to securely deregister my account under specific conditions.
- As an organization, I would like to view organization managers/registered employees.

2. User information security

- As a user, I would like to be notified of where my information goes.
- As a user, I would like to have my information stored securely.
- As an organization/organization manager, I would like to have my information stored securely.

3. User specific interface

- As a user, I would like to have the option of having a donor-oriented screen or receiver-oriented screen.
- As a user, I would like to update my donor-oriented status or receiver-oriented status.

4. Item requesting and donating

- As a receiver-oriented user, I would like to order items already in the catalog.
- As a receiver-oriented user, I would like to request items not in the catalog.
- As a donor-oriented user, I would like to fulfill user requests.
- As a donor-oriented user, I would like to see local requests.

5. Item delivery/reception

- As a receiver-oriented user, I would like to check an item’s delivery status.
- As a donor-oriented user, I would like to deliver

items for organizations.
- As a donor-oriented user, I would like to have the option to donate items directly to receivers at a public location, at the donor's address, or at the donee's address. Either face-to-face or left on the doorstep of the specified address.
- As a receiver-oriented user, I would like to have the option to receive items directly to receivers at a public location, at the donor's address, or at the donee's address. Either face-to-face or left on the doorstep of the specified address.

6. Non-profit specific donating

- As a donor-oriented user, I would like to be able to donate items to non-profit organizations.
- As a non-profit organization, I would like to have a profile listing my locations, mission statement, and logo.
- As a user, I would like to view non-profit organization profiles.

7. Stock Updation

- As an organization/organization manager, I would like to add items to my inventory.
- As an organization/organization manager, I would like to delete items in my inventory.
- As an organization/organization manager, I would like to edit items in my inventory.

8. Item searching/browsing

- As an organization/organization manager, I would like to search for items by category in my inventory.
- As an organization/organization manager, I would like to search for items by name in my inventory.
- As an organization/organization manager, I would like to search for items by reference number in my inventory.
- As a user, I would like to search for items by category in the catalog.

- As a user, I would like to search for items by name in the catalog.
- As a user, I would like to browse items.

9. Demand forecasting

    - As an organization/organization manager, I would like to utilize demand forecasting to manage inventory supplies.

10. Item categorization

    - As an organization/organization manager, I would like to make custom hashtags/custom categories.
    - As a donor-oriented user, I would like to make custom hashtags/custom categories for items.

Non-Functional Requirements

- Appearance
    - As a developer, I would like the appearance to be pleasing to use
    - As a developer, I would like to have tabs to guide the user to where they need to go
    - As a developer, I would like the user to be able to customize their interface using filters for location, needs, and interests
- Security
    - As a developer, I would like to have each user have a specific login
    - As a developer, I would like for users to not be able to access accounts they are not authorized to access
    - As a developer, I would like for users to choose whether or not they want to disclose their location
    - As a developer, I would like for an organization to have multiple accounts, each with separate roles and permissions
- Performance
    - As a developer, I would like the application to spend

    less than 10 seconds loading each new page
  - As a developer, I would like the application to run without unexpected crashes
- Server
  - As a developer, I would like the server to update in real-time
  - As a developer, I would like the server to store user information and retrieve it easily and without flaws
  - As a developer, I would like the application to support at least 1000 users
- Design
  - As a developer, I would like the application to be intuitive for users
  - As a developer, I would like for the application to run on the web
  - As a developer, I would like for the UI and the backend to communicate efficiently using APIs
  - As a developer, I would like for any errors to be handled gracefully

Design Outline

High Level Overview

Charity Link will be a web application that allows organizations to manage their inventory and streamline their donation and delivery process; donees to easily receive donations from organizations and donors as well as request items; and donors to see what is needed by organizations and donees. Charity Link will utilize the client-server model where a server will handle the API requests from all the UIs, built on React, using the Java Spring Boot framework. The server will then retrieve and modify data from the relational database, using SQL and the Hibernate framework, according to the API calls from the UI.

1. Client
    a. Client creates the GUI for the user to interact with the product
    b. Client makes API calls to the server
    c. Client receives API data (likely in JSON format)
2. Server
    a. Server receives API calls from clients
    b. Server makes queries to the database to modify/retrieve data
    c. Server generates (most likely JSON) responses and sends them to the corresponding clients
3. Database
    a. Relational database to store all of the raw data necessary for the application to carry out its functional requirements
    b. Database modifies/retrieves data based on the queries made by the server

Design Issues

Functional Issues:

1. What information do we need for signing up for an account?
    - Option 1: Username and password only
    - Option 2: Username, email address, and password

    Choice: Option 2

    Justification: In terms of setting up user accounts, it is generally considered more secure and effective to gather an email address alongside having their username and password. Having an email address associated with the account provides an additional layer of security, as well as a means of resetting the password in the event that the user forgets it. Additionally, an email address can be used for account verification, especially for organizations to register their accounts. Thus having an email address on top of having their username and password ensures that the user's account is secure and can be easily managed and accessed. Location and Name are optional and can both be updated in settings and when making requests.

2. How do we categorize inventory items?
    - Option 1: Alphabetical or numerical categorization
    - Option 2: Hashtagging items

    Choice : Option 2

    Justification: Categorizing items using hashtagging is more flexible and adaptable than alphabetical categorization since it doesn’t bind items to stick to one of the existing categories while keeping intact the essence of type-based categorization. With hash tagging, users can add as many tags as they need to each item, allowing them to categorize items in multiple ways and viewing them from different angles. This is particularly useful for categorizing items that belong to multiple categories. Furthemore, hashtagging is intuitive and user-friendly. It's a system that most people are familiar with, making it easy to understand and use. Lastly, hashtagging makes it easier to search for items within your inventory. By using relevant tags, users can quickly find items based on their category, making it more efficient and faster to manage inventory.

3. What is the best way to get items delivered to the people in need?
    - Option 1: At the donor’s specified address
    - Option 2: At the receiver’s specified address
    - Option 3: At a commonly decided public place by both the donor and the receiver
    - Option 4: All of the aforementioned options

   Choice: Option 4

   Justification: The best way to get items delivered to people in need is by using Option 4 because it allows for flexibility and adaptability, which is crucial in ensuring that the items get delivered to the right people as quickly and efficiently as possible. For example, if the donor is unable to drop off the items at the receiver's specified address, a commonly decided public place may be a better option. On the other hand, if the receiver is unable to travel to pick up the items, having them delivered directly to their specified address might be the best choice. This approach is similar to the delivery system used by Facebook Marketplace, which allows users to choose from various delivery options, including local pickup or public meetup. Thus offering all of the delivery options gives the users a level of flexibility that ensures they can get the items they need, regardless of their location or availability.

4. Whom does a donor donate items to, so that they reach the people in need?
    - Option 1: Directly to the people in need
    - Option 2: To nonprofits that hold specific outreach programs to give items to the people in need

   Choice: Options 1 and 2

   Justification: Choosing both options 1 and 2 offers donors the flexibility to choose the method that works best for them. If the donor would like to have a direct impact, they can choose to donate directly to the people in need. On the other hand, if the donor would like to support a specific cause, they can choose to donate to a nonprofit that holds specific outreach programs to give items to the people in need. Offering both options, can help cater to a wider range of donors, ensuring that the platform is accessible and useful to all. To implement this, our platform will give donors not only a catalog of nonprofits but also a list of donee requests to choose which ones to donate to.

5. What is the best way to employ demand forecasting for our platform?
    - Option 1: Rule-based forecasting
    - Option 2: Data analytical forecasting
    - Option 3: Intuition-based forecasting

   Choice: Option 2

   Justification: For nonprofits, the best method to employ demand forecasting would be by using data analytical techniques to forecast their demand. By analyzing data from past donations, the organizations can identify patterns and trends, and use this information to make informed predictions about future demand. This can help nonprofits better plan their outreach programs and ensure that they have enough resources to meet the needs of the people they serve. This is a more efficient and effective approach than rule-based or intuition-based forecasting, which can often be limited by the knowledge and experience of the individual making the predictions. Thus, by giving them a medium to analyze their previous donation data, nonprofits can forecast their demand in a more accurate and reliable way.

Non-Functional Issues:

1. What cloud service should we host our API/Database
    - Option 1: Azure
    - Option 2: AWS
    - Option 3: Google Cloud
    - Option 4: Firebase

   Choice: Azure
   Justification: Considering the scope of our application the cloud server that we host our API/Database on has little effect on our product. We are primarily looking for a PaaS and while Azure is more of an IaaS since, as Purdue students who have Microsoft accounts, we get a year of Azure for free we decided that it would be most cost effective in the developmental stages of our application.

2. What backend language/framework should we use for the server/API?
    - Option 1: Spring Boot/Java
    - Option 2: PHP
    - Option 3: Node.js/JavaScript
    - Option 4: Django/Python

   Choice: Spring Boot/Java

   Justification: One of our primary motivations for choosing Spring Boot and more importantly Java is that our team is very familiar with Java especially considering the experience that we have from CS 180. Furthermore, we chose Spring Boot as it is a relatively simple framework to create a RESTful API in Java that can easily interact with a database.

3. What web development language/framework should we use?
    - Option 1: React/JavaScript
    - Option 2: Angular/TypeScript
    - Option 3: Vue/JavaScript
    - Option 4: HTML + JavaScript

   Choice: React

   Justification: Firstly, our team has the highest proficiency in JavaScript. Secondly, we wanted to use a framework rather than coding out raw HTML and JavaScript. This already left us with React and Vue. Ultimately, we chose React as it is open source and there is more access to community made third-party tools.

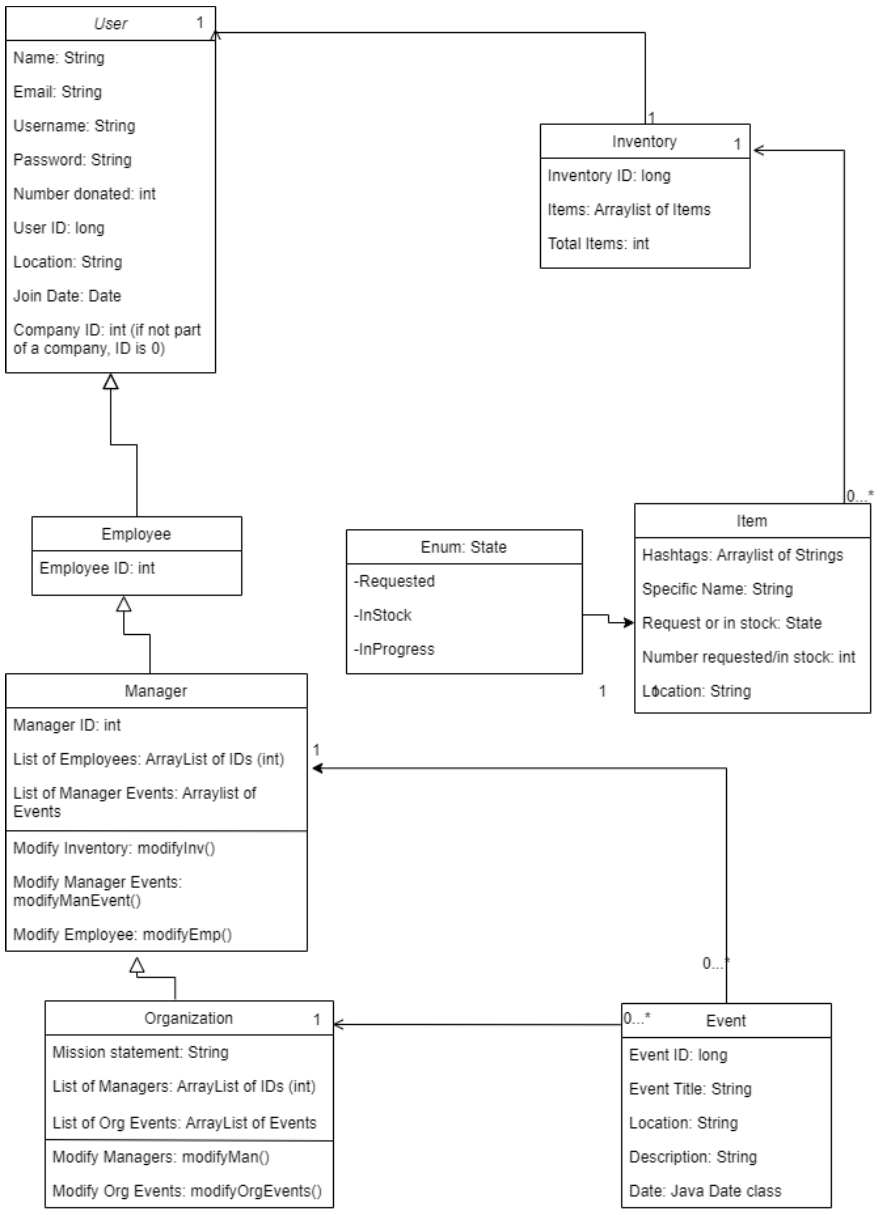
4. What database should we use?
    - Option 1: MySQL
    - Option 2: Oracle
    - Option 3: Microsoft Access
    - Option 4: MongoDB

   Choice: MySQL

   Justification: Firstly, our team has the highest proficiency in SQL which would make the research time for this database the shortest. Secondly, MySQL is an extremely secure and portable relational database that can effectively store all of our necessary data.

Design Details

Class Design



User
1
Name: String
Email: String
Username: String
Password: String
Number donated: int
User ID: long
Location: String
Join Date: Date
Company ID: int (if not part of a company, ID is 0)
1
Inventory
1
Inventory ID: long
Items: Arraylist of Items
Total Items: int
Employee
Employee ID: int
Enum: State
-Requested
-InStock
-InProgress
0...*
Item
Hashtags: Arraylist of Strings
Specific Name: String
Request or in stock: State
Number requested/in stock: int
Location: String
1
1
Manager
Manager ID: int
List of Employees: ArrayList of IDs (int)
List of Manager Events: Arraylist of Events
Modify Inventory: modifyInv()
Modify Manager Events: modifyManEvent()
Modify Employee: modifyEmp()
1
0...*
Organization
1
Mission statement: String
List of Managers: ArrayList of IDs (int)
List of Org Events: ArrayList of Events
Modify Managers: modifyMan()
Modify Org Events: modifyOrgEvents()
0...*
Event
Event ID: long
Event Title: String
Location: String
Description: String
Date: Java Date class

Description of Classes and Interactions

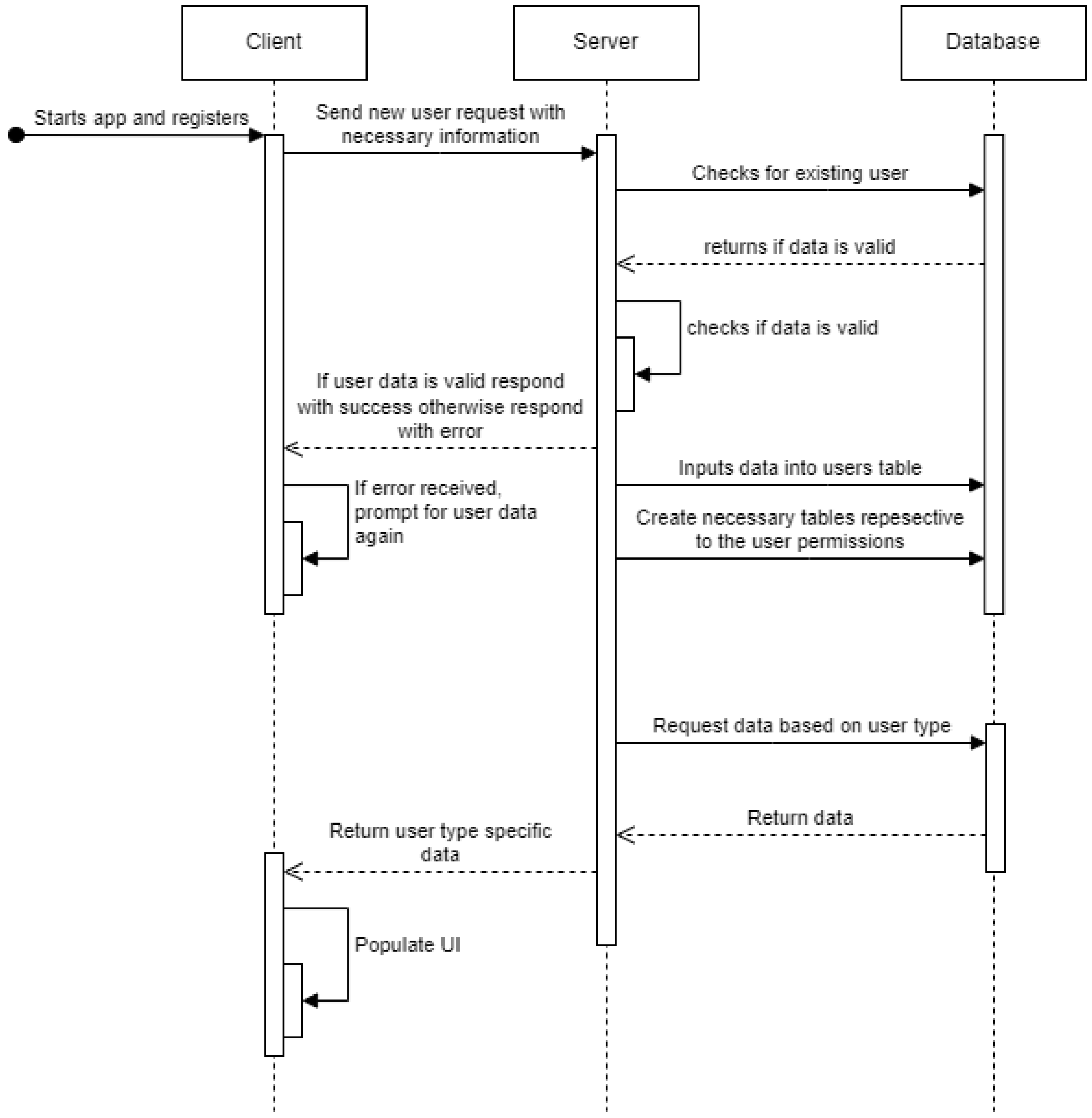
- USER Superclass
    - User is created when someone registers
    - Contains a username, a password, a user ID, a name, and an email
    - Contains optional information such as location
    - Contains a list of items that they have requested
    - Contains a list of inventory items they have that can be donated
    - Contains a list of donations they have completed
    - MANAGER Subclass
        - Contains high level permission that can access, control, and modify employees and items
        - ORGANIZATION subclass
            - Contains a company ID, a mission statement
            - Contains list of employees and managers
- Inventory
    - The inventory class contains the inventory for all of the items a company or user needs and has in stock
    - Contains an inventory ID
    - Contains a list of items
    - Contains an integer with the number of items in the inventory
- Item
    - An arraylist of hashtags in the form of strings where the user can describe what the item is using hashtags (for example, for a toothbrush someone could put the tags dentistry, toiletry, toothbrush, hygiene)
    - User inputs specific name of item
    - Contains an item ID
    - Contains an enum on whether the item is being requested, in stock, or in the process of being delivered
    - Contains an integer with the number of items in stock / is being requested

- Event
    - Contains an Event ID
    - Contains an event title
    - Includes location of event
    - Includes description of event
    - Includes date of Event

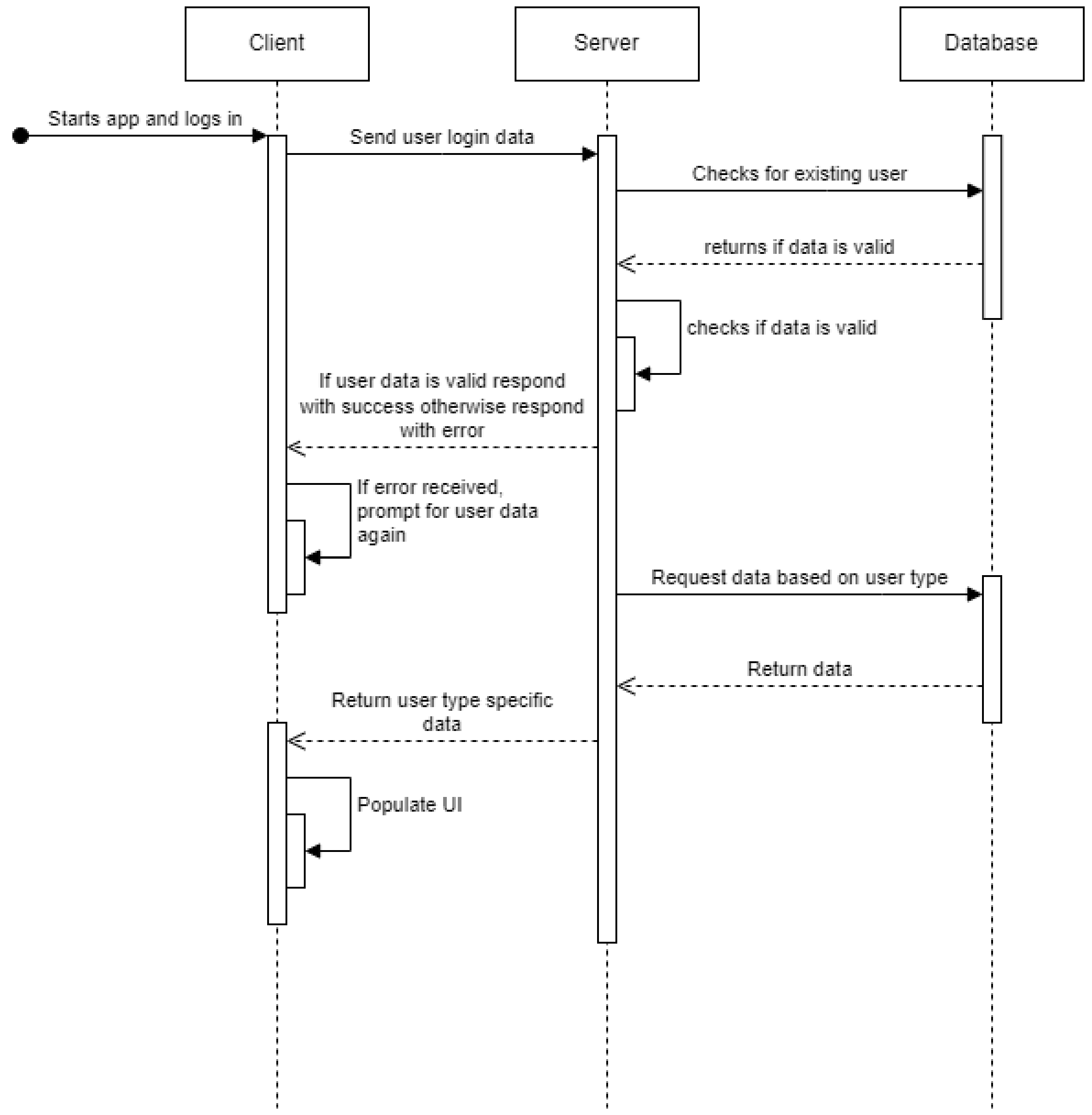
Sequence Diagrams

The diagrams below illustrate the progression of key events in the application, starting with a user creating a profile, followed by logging in, an organization viewing and altering inventory, users viewing requests from recipients and organizations, and finally, the user creating a request. These sequence diagrams demonstrate the communication process in a client-server model. When a user interacts with the frontend interface, the client sends a request to the server. The server then queries the database to retrieve the necessary data. After processing the data, the server sends a response back to the client. Finally, the client processes the data and updates the display on the UI.

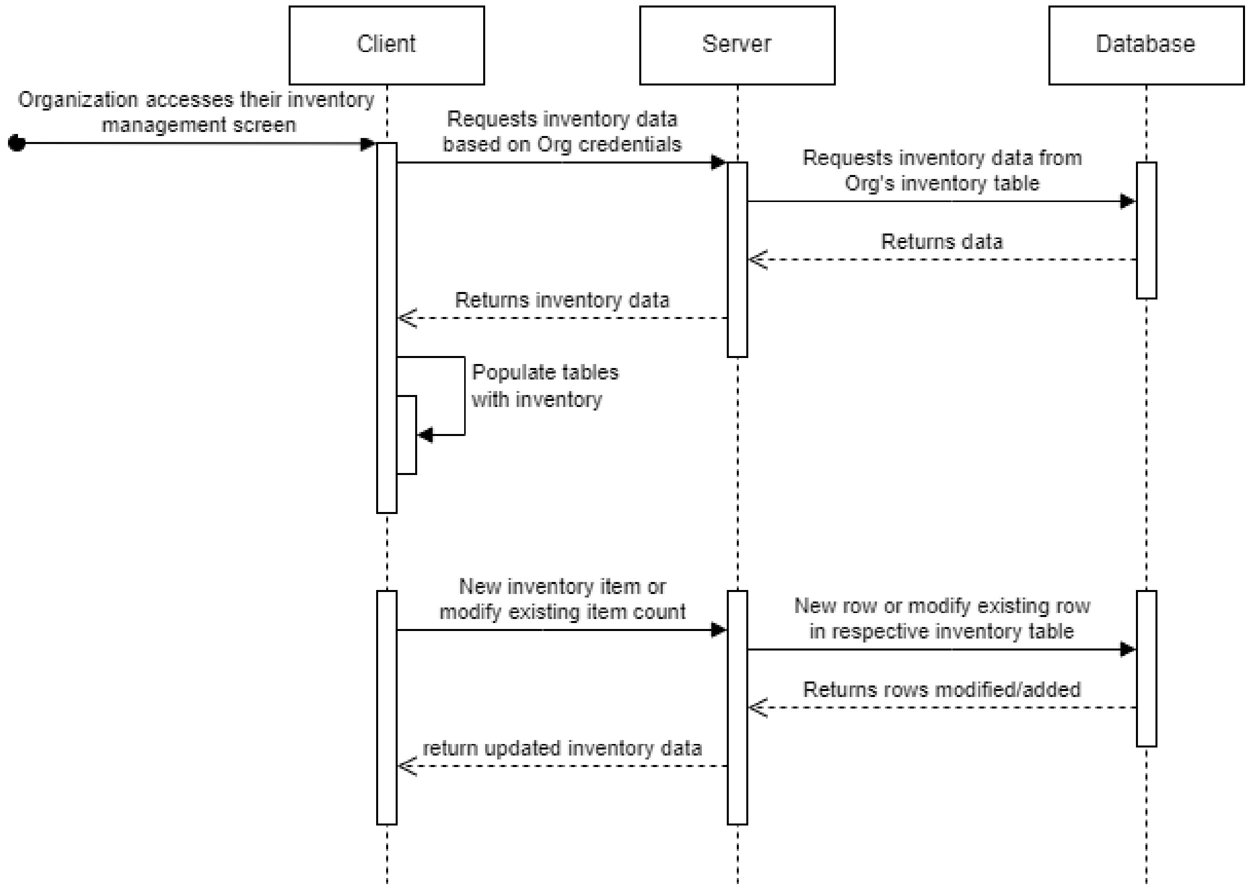
1. User creates a profile


Client
Server
Database
Starts app and registers
Send new user request with necessary information
Checks for existing user
returns if data is valid
checks if data is valid
If user data is valid respond with success otherwise respond with error
Inputs data into users table
If error received, prompt for user data again
Create necessary tables repesective to the user permissions
Request data based on user type
Return data
Return user type specific data
Populate UI

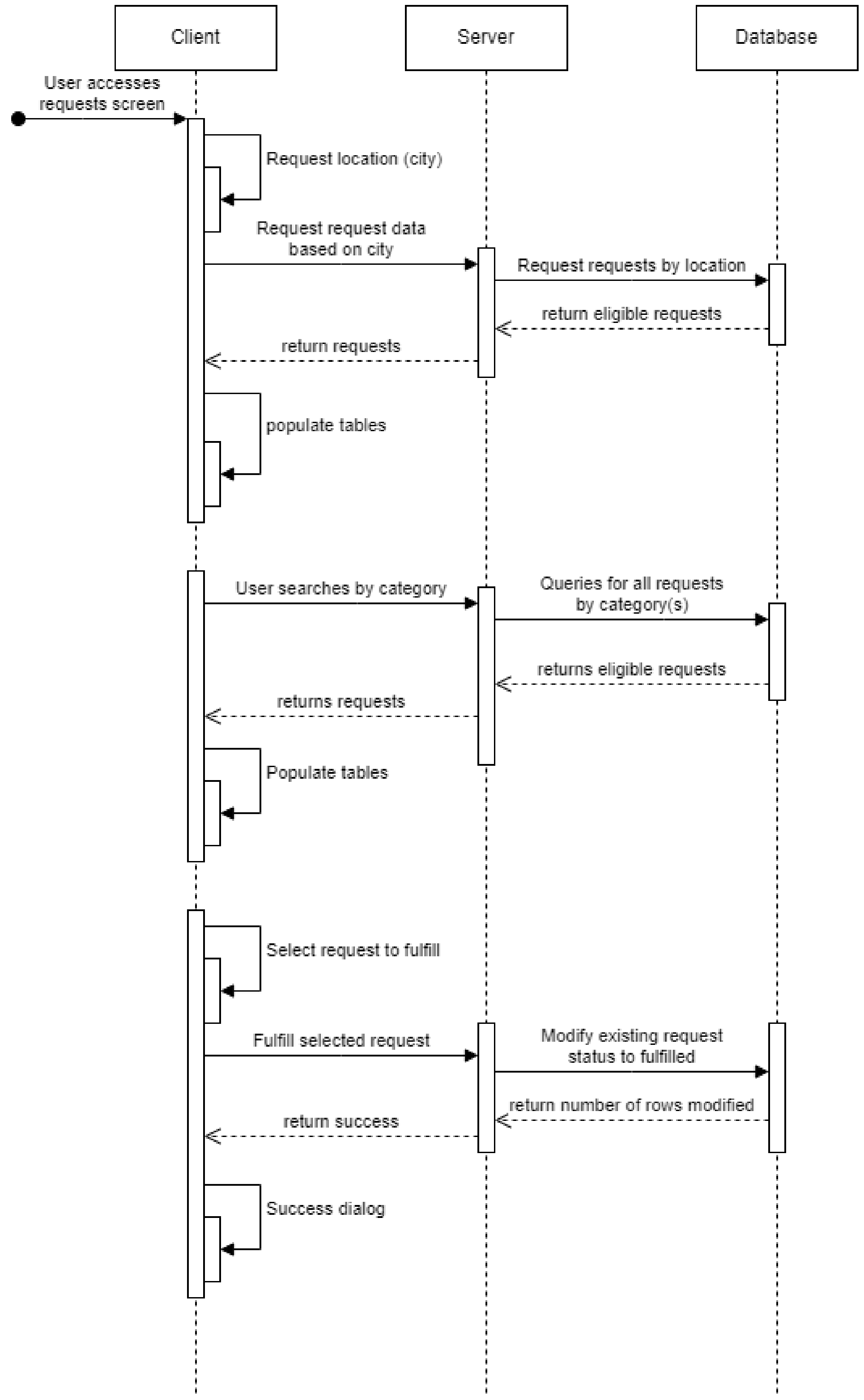
2. User login


Client
Server
Database
Starts app and logs in
Send user login data
Checks for existing user
returns if data is valid
checks if data is valid
If user data is valid respond with success otherwise respond with error
If error received, prompt for user data again
Request data based on user type
Return data
Return user type specific data
Populate UI

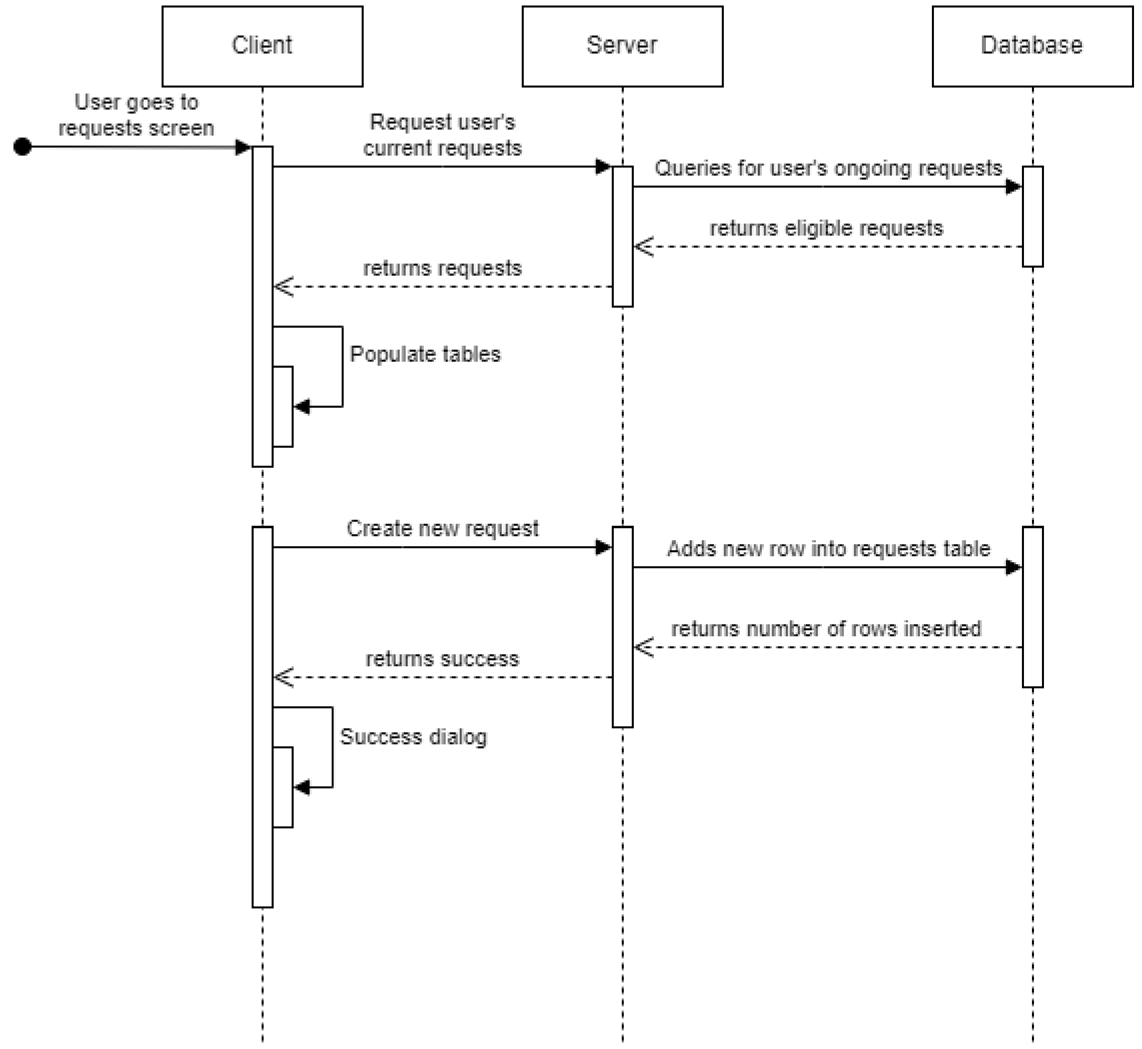
3. Organization viewing and modifying inventory


Client
Server
Database
Organization accesses their inventory management screen
Requests inventory data based on Org credentials
Requests inventory data from Org's inventory table
Returns data
Returns inventory data
Populate tables with inventory
New inventory item or modify existing item count
New row or modify existing row in respective inventory table
Returns rows modified/added
return updated inventory data

4. Users viewing requests from donees and organizations


Client
Server
Database
User accesses requests screen
Request location (city)
Request request data based on city
Request requests by location
return eligible requests
return requests
populate tables
User searches by category
Queries for all requests by category(s)
returns eligible requests
returns requests
Populate tables
Select request to fulfill
Fulfill selected request
Modify existing request status to fulfilled
return number of rows modified
return success
Success dialog

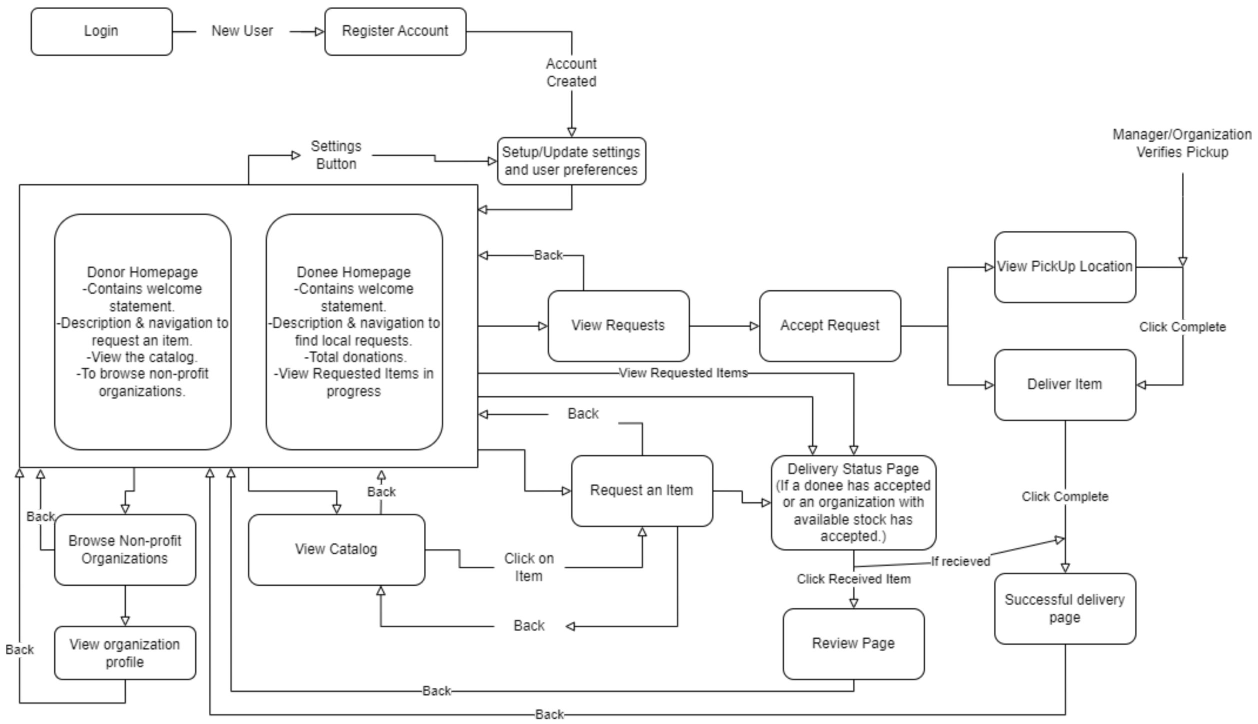
5. User creating request


Client
Server
Database
User goes to requests screen
Request user's current requests
Queries for user's ongoing requests
returns eligible requests
returns requests
Populate tables
Create new request
Adds new row into requests table
returns number of rows inserted
returns success
Success dialog

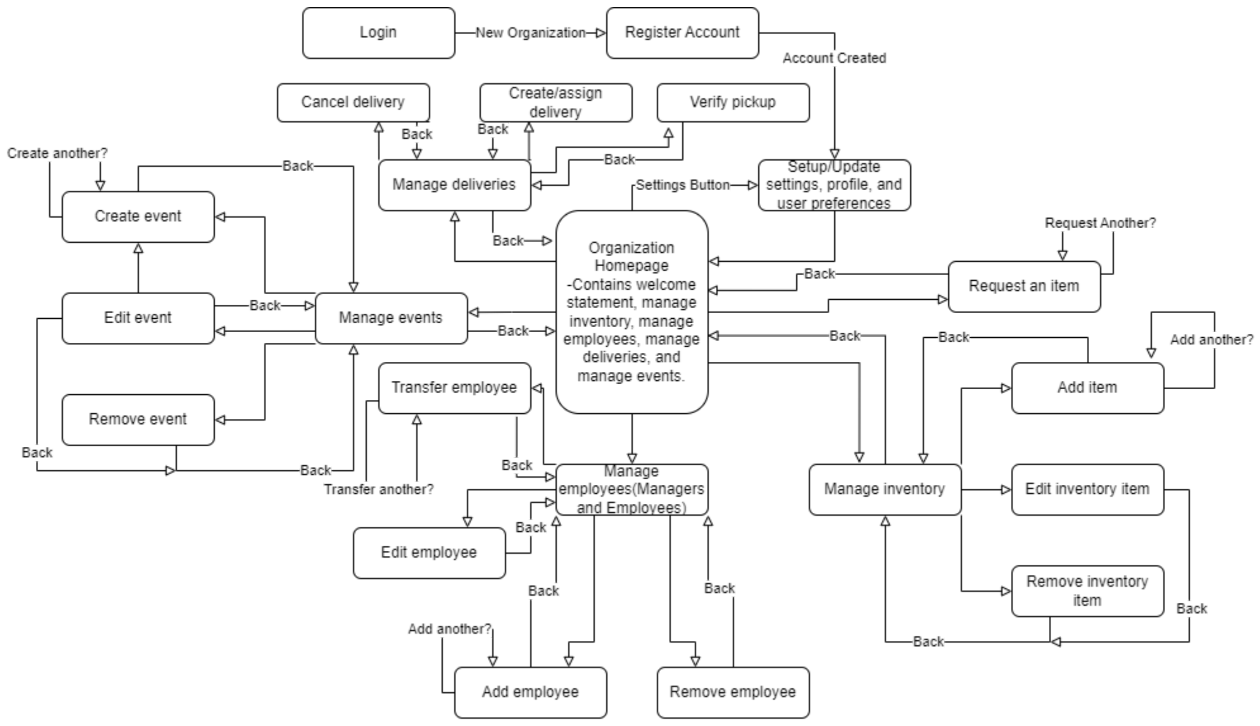
Navigational Flow Maps

After logging in with a company or as an average user, we wanted to ensure that all options are present on the main page for ease of access to all users. All functionalities will also be accessible in the sidebar dropdown menu. There will be a welcome statement and number of donations, if relevant. A settings button will lead all users to where they can update their information and organizations can update their profiles in this section. Users can view companies or the catalog by searching by name or by hashtags. The logo on the top of the homepage will always go back to the homepage, but there will also be a back button. Events can be viewed and will have four categories: Title, Description, Date, and Location.

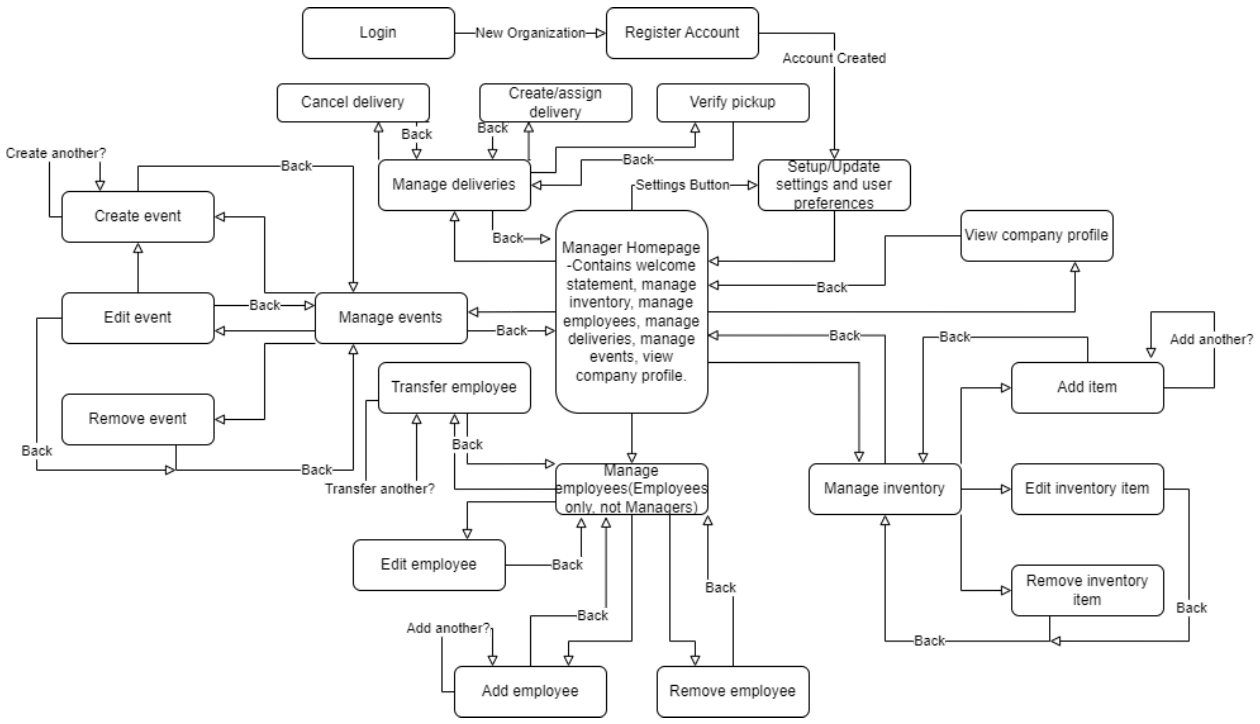
1. Donee and Donor Navigation Map


Login
New User
Register Account
Account Created
Settings Button
Setup/Update settings and user preferences
Manager/Organization Verifies Pickup
Donor Homepage -Contains welcome statement. -Description & navigation to request an item. -View the catalog. -To browse non-profit organizations.
Donee Homepage -Contains welcome statement. -Description & navigation to find local requests. -Total donations. -View Requested Items in progress
Back
View Requests
Accept Request
View PickUp Location
Click Complete
Deliver Item
View Requested Items
Request an Item
Delivery Status Page (If a donee has accepted or an organization with available stock has accepted.)
Browse Non-profit Organizations
View Catalog
Click on Item
If recieved
Click Received Item
Successful delivery page
Review Page
View organization profile

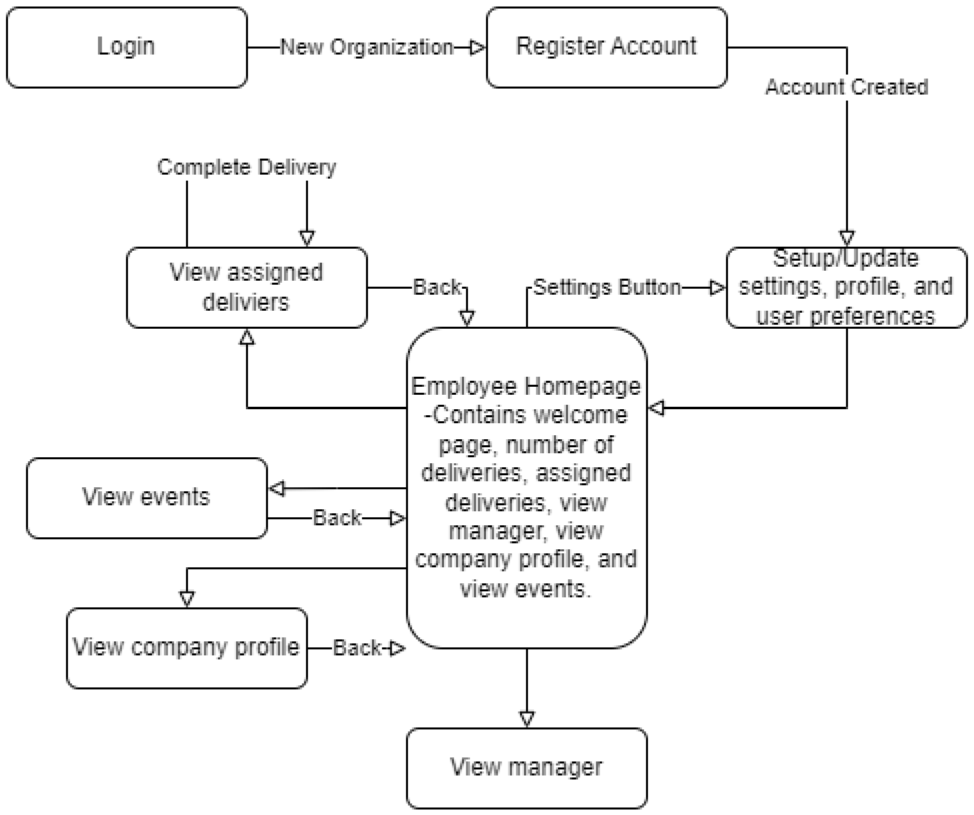
2. Organization Map



3. Manager Map

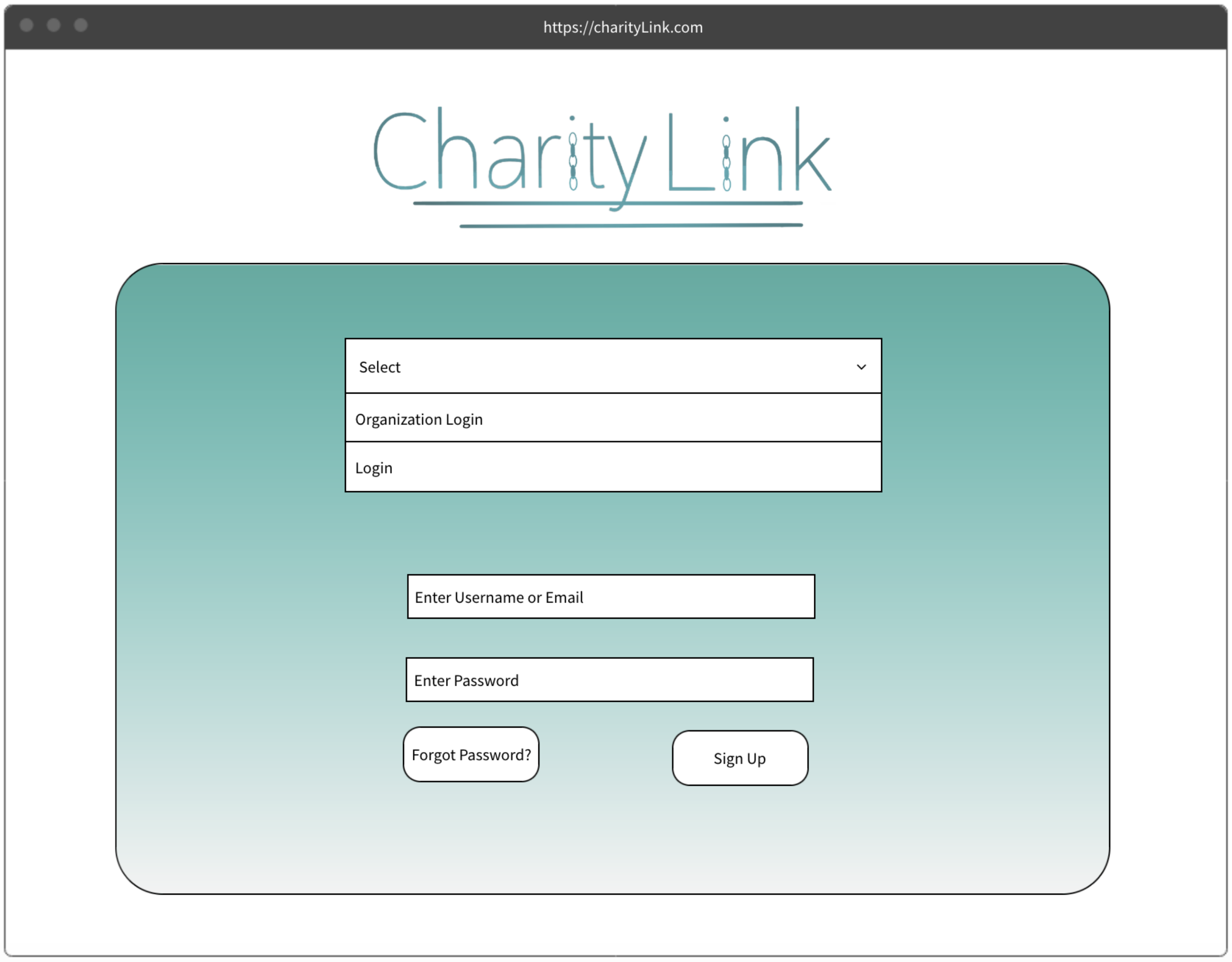


4. Employee Map

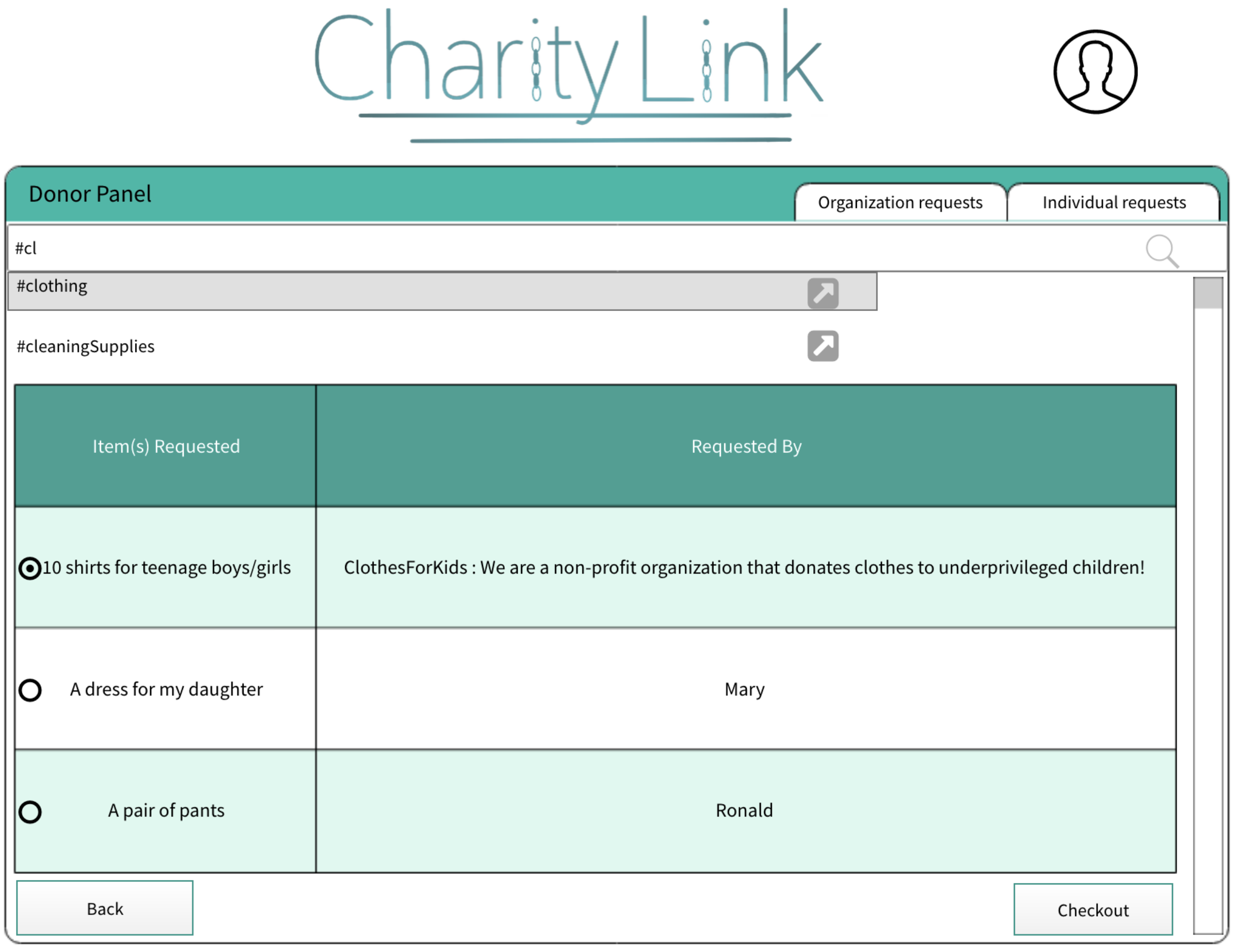

Login
New Organization
Register Account
Account Created
Complete Delivery
View assigned deliviers
Back
Settings Button
Setup/Update settings, profile, and user preferences
Employee Homepage -Contains welcome page, number of deliveries, assigned deliveries, view manager, view company profile, and view events.
View events
Back
View company profile
Back
View manager

UI Mockups

The following are rough UI mockups for CharityLink. We decided to create a login screen and the display of the Donor Panel. We also settled on a color palette for our website.



This is the login page. It has a drop down menu to select from depending on if the user is an organization manager or an individual user. It has a button for the users to reset their password in case they forgot it. It also has a button for users to sign up instead if they don’t have an existing account.



Item(s) Requested	Requested By
[✓] 10 shirts for teenage boys/girls	ClothesForKids : We are a non-profit organization that donates clothes to underprivileged children!
A dress for my daughter	Mary
A pair of pants	Ronald

This is the donor panel which gives a donor-oriented user a list of requests from both individuals and non-profit organizations based on the category that they search to donate for. They additionally have the option of sorting the requests specifically by organizations or specifically by individuals. They can scroll through these requests and choose the one that they want to fulfill and checkout that item. They can also go back to the previous page as well as click on their user icon to manipulate their user settings.